Future Generation Computer Systems 54 (2016) 16–40

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Web-centred end-user component modelling

David Lizcano [a,*], Fernando Alonso [b], Javier Soriano [b], Genoveva López [b]

[a] Universidad de Distancia de Madrid (UDIMA), Spain
[b] Universidad Politécnica de Madrid, Spain

HIGHLIGHTS

- Programming-illiterate users face barriers exploiting web service composition tools.
- We offer a user-centric approach to Internet of Services to tackle these obstacles.
- This approach adapts service front-ends for end users to build SOA-based software.
- Thus, end users can create applications to support their routine work on their own.
- The presented approach elicits the best practices and principles of the current SOTA.

ARTICLE INFO

Article history:
Received 2 September 2014
Received in revised form
15 June 2015
Accepted 1 July 2015
Available online 10 July 2015

Keywords:
End-user programming
Web engineering
Component-based software
Human factors
Visual programming
Component modelling

ABSTRACT

This paper formally defines a web component model enabling end-user programmers to build component-based rich internet applications (RIAs) that are tailored to meet their particular needs. It is the product of a series of previously published papers. The formal definition in description logic verifies that the model is consistent and subsumes currently existing models. We demonstrate experimentally that it is more effective than the others.

Current tools propose very disparate web component models, which are based on the appropriate invocation of service backends, overlooking user needs in order to exploit these services and resources in a friendly manner. We have proposed a web model based on a detailed study of existing tools, their pros and cons, limitations and key success factors that have enabled other web end-user development (WEUD) solutions to help end-user programmers to build software to support their needs. In this paper we have verified that the proposed model subsumes and is instantiated by the models of the other existing tools that we analysed, coming a step closer to the standardization of end-user centred RIAs and development environments. We have implemented a development tool, called EzWeb, to produce RIAs that implement the proposed model. This tool enables users to develop their application following the model's component structure based on end-user programming success factors. We report a statistical experiment in which users develop increasingly complex web software using the EzWeb tool generating RIAs that conform to the proposed component model, and other WEUD tools generating RIAs that conform to other models. This experiment confirms the applicability of the proposed model and demonstrates that more end-user programmers (EUPs) (users concerned with programming primarily for personal rather public use) successfully develop web solutions for complex problems using the EzWeb tool that implements the model, which is more efficient than existing tools that implement other models.

1. Introduction

Interest and investment in web end-user development (WEUD) are mounting all the time, and its impact [1] has even outstripped forecasts made by Christopher Scaffidi, Brad Myers and Mary Shaw back in 2005 [2]. There are many web-based mashup development environments that enable millions of users to personally develop software solutions to solve their own problems.

Many software suppliers including Microsoft, Apple, IBM, Yahoo!, Oracle, etc., have developed tools providing support for end-user programmers (EUPs) (programmers who wish to achieve the result of a program primarily for personal rather public use) [1] to develop web applications, particularly rich internet applications (RIAs), offering do-it-yourself (DIY) [3] guidance on how to

* Corresponding author.
E-mail address: david.lizcano@udima.es (D. Lizcano).

http://dx.doi.org/10.1016/j.future.2015.07.002
0167-739X/

evolve end-user developments to meet end-user demands and requirements. Such applications include Chrome Web Store (Chrome WS) and its Developer Tools [4], Yahoo! Pipes and Dapper [5,6], Microsoft Popfly [7] (currently closed and offered as part of Microsoft WebMatrix), Kapow Platform [8], JackBe Presto [9], AMICO Sketchify [10], Marmite [11] or EzWeb [12].

These solutions enable EUPs to develop their own software solutions. These solutions help EUPs create a graphical user interface (GUI) by visually connecting components with different levels of abstraction in order to access and exploit different types of services and resources and solve their particular problem. Each solution has pros and cons [3,1] and offers distinctive WEUD functionalities for creating end-user solutions. The major weakness is that each tool defines a web application development model for building solutions to problems of a particular type and complexity. These models are of no use for EUPs to develop more complex general-purpose RIAs [13,14]. For example, Yahoo! Pipes is confined to building mashups of data from RSS or HTML sources, whereas Kapow Platform specializes in building web portals using screen scraping techniques, and so on. The important thing, though, is that these WEUD solutions are promoting a new web component model [15] that has not yet, however, been either fully structured or formalized. The component models used in these tools, their strengths and generated products have not yet been studied in detail in order to define a comprehensive component model for the web. The race to compete in an increasingly globalized WEUD solutions ecosystem has forced developers (Google, Yahoo!, Microsoft, Amazon, Apple, Sun, IBM, etc.) to develop and optimize their own tools in their application environments without formalizing a common underlying component model. Therefore, a common component model needs to be built in order to promote interoperability between building blocks supplied by different manufacturers [16] and raise acceptance among EUPs by guaranteeing that users can successfully build more complex general-purpose RIAs than they can now [17].

The challenge, then, is to come up with an emerging web end-user component model [18] that covers the functionalities of a well-known set of existing tools, exploits their strengths and, whenever possible, reduces their weaknesses, encouraging EUPs to create and/or customize their own software [19]. This paper studies a representative set of existing tools, which were selected as being the most commonly used and successful tools in recent years, analyses the component models underlying the RIAs created using each tool and defines and formalizes in description logic a component model that subsumes the RIA models by merging their functionalities and strengths and incorporating EUD (end-user development) success factors. A WEUD tool that instantiates this model has been tested on real EUPs and found to more effectively scale up to increasingly complex problems than today's EUD tools. We designed this tool, called EzWeb [20], along with other partners under the auspices of a Networked European Software and Service Initiative (NESSI) strategic research project. EzWeb is now being used in two European Union 7th Framework Programme projects in which we are participating: 4CaaSt [21] (building the future Platform as a Service) as part of its mashup-as-a-service solution and FI-WARE [22] (building the Future Internet core platform) as part of its applications and services ecosystem and delivery framework's generic enablers for EUPs to build application mashups.

The remainder of the paper is structured as follows. Section 2 presents related work and analyses the principal WEUD tools and the component models governing the end-user solutions that they can each build. Section 3 presents a set of target features for an end-user oriented component model and presents our WEUD component model that combines the strengths of the other models with EUD success factors that we have analysed during our research. This model has been mathematically formalized in Section 4 using formal logic to demonstrate that it is consistent and is instantiated by the models produced by the analysed WEUD tools. Section 5 describes the use of an automatic reasoning tool to check whether the component model generated by each tool described in Section 2 is a valid instance of the global model reported in Section 3. Section 6 presents the results of a study that we conducted to test whether EzWeb, which generates RIAs that conform to the proposed component model, achieves better results than other WEUD tools, which generate RIAs that conform to other models. Section 7 addresses the EUD dilemma of whether it is better to define generic or domain-specific EUD tools. Finally, Section 8 concludes this paper and presents a brief outline of future work.

2. Related work: existing solutions for end-user development

Software suppliers are in the process of converting their products into web services (an approach termed Software as a Service, SaaS), and all sorts of software solutions are readily available in the shape of services scattered over the Internet [23]. These approaches target end users that are generally unfamiliar with the details of the technology used to implement services. Users should now be just as able to use these services to their own advantage as they used to be able to use commercial software products in the past [24]. There are compilations of available services, together with examples, guidelines and success stories in service use, including the Programmable Web repository [25]. Programming knowledge, knowledge of SOAP, WSDL, BPEL, etc., is required to use these resources [26]. This breach between the high availability of web resources and the low prospects of their use by EUPs has led many large software enterprises to create mashup development environments targeting EUPs like Chrome WS and its Developer Tools, Yahoo! Pipes and Dapper, Microsoft Popfly, Kapow Platform, JackBe, AMICO, Marmite or EzWeb. They all share the goal of enabling EUPs to develop a composite web application that solves their particular problem.

The major problem with these tools is that EUPs are often unable to translate their particular requirements into a specific software product [1,17], because each tool focuses on achieving a particular solution type that does not necessarily meet user needs. For example, Yahoo! Pipes creates a correctly filtered data list feed, Kapow Platform creates an execution flow based on pre-existing interlinked web portals, and so on. Users who require a more complex RIA or need to solve a problem type other than for which the tool was designed will be disappointed.

Our working hypothesis is that the component models controlling the different WEUD tool solutions are not general enough to be able to create more complex general-purpose RIAs. Additionally, the tools do not match the way in which EUPs conceive their solution; nor do they offer a natural development process for end-user characteristics and needs. This hypothesis is based on the study of many related papers focusing on the EUD field and applicable to WEUD, which are described below.

End-user development or EUD is a term first proposed by European researchers ten years ago at an international symposium held in Bonn, Germany. It has attracted a lot of scientific interest since the first biannual International Symposium on End-User Development (IS-EUD) focusing on this domain was held in 2007. Four top-level meetings have been organized since then. The main topic of these conferences is how to empower EUPs to develop and adapt systems themselves.

These symposiums, together with other international congresses, have promoted several lines of EUD-related investigation akin to the research reported in this paper: (1) attempts at simple programming languages or environments focused on a particular domain, such as EnglishMash (an end user-oriented language